Code EE2L21

KITT Datasheet

Project Autonomous Driving Challenge

April 25, 2019

Delft University of Technology

M.E. Weskin
J.H. Belier and J. Martinez (editors)

TU Delft
Delft University of Technology

Challenge the future

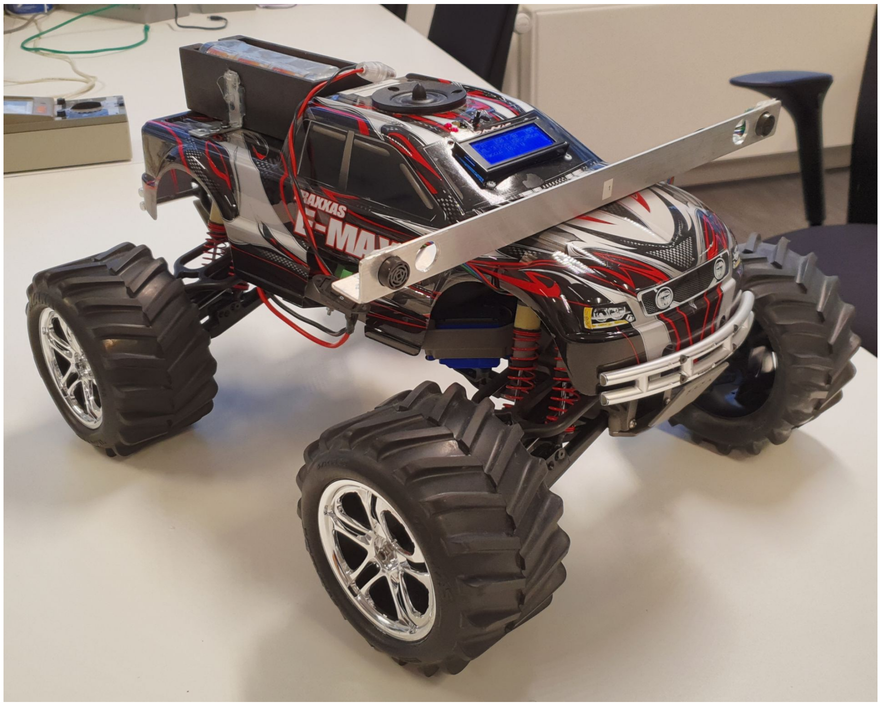


Figure 1: KITT: A smart, autonomous electric car.

1 Overview

In this datasheet KITT (Knowledge & Intelligence Twenty Twenty) and all its features are described. KITT —named like this for nostalgic reasons— is an autonomous driving electric car which can locate itself and detects objects to prevent a collision. KITT is equipped with an pre-programmed controller that converts the control signals given via the Bluetooth communication.

2 Features

KITT is build for outside use on rough terrain and to operate autonomously. The main features are listed below:

1. 32-bit NXP LPC4357 microcontroller with ARM CortexTM-M4/M0 processor running at 72MHz to control the car and the audio beacon (the MCU is not directly accessible).

2. Texas Instruments LM4752 11 W audio power amplifier for the audio beacon.

3. Rechargeable battery, 16.8 V nominal.

4. Texas Instruments LM2676 20 V to 5 V DC-DC converter for internal operating voltages.

5. Bluetooth module for serial communication with the base station. Via this serial communication the car is controllable.

Table 1: Absolute Maximum Ratings

Battery voltage	22 V
Battery output current	Internally limited with fuse (10 A, slow)
Motor controller input voltage	20 V
MCU I/O pin voltage	3.3 V
Velocity	75 km/h (internally limited to ca. 20 km/h).

6. LCD Display with information about the status of the car.

7. SRF02 ultrasonic sensors for distance measurement.

3 Specifications

The absolute maximum ratings are listed in Table 1. The output current of the battery is high (order of tenths of Amps), so it is internally limited with a slow fuse of 10 A.

The car is sensitive to external influence; that is why a solid control system is required on the host system. The controller MCU's operating voltage is 3.3 V, which means that the voltage on its I/O-pins is at most 3.3 V. The car is built for outdoor use on free, rough terrain. For safety reasons the velocity is internally limited.

The operating ratings you have to determine by yourself. Fill them in, in Table 2, to complete the datasheet.

Table 2: Operating Ratings

Bluetooth operating frequency	...
Bluetooth operating range	...
Ultrasonic sensor resolution	...
Ultrasonic sensor distance	...
Turning radius	...
Max. Command refresh time	...

4 Application Info

The car contains several control PCB's. A schematic diagram of the entire system is shown in Figure 2. The green blocks are PCB's on the car, the blue ones are microcontrollers and the red blocks are hardware peripherals (i.e. sensors and actuators). Important parts are described in this section.

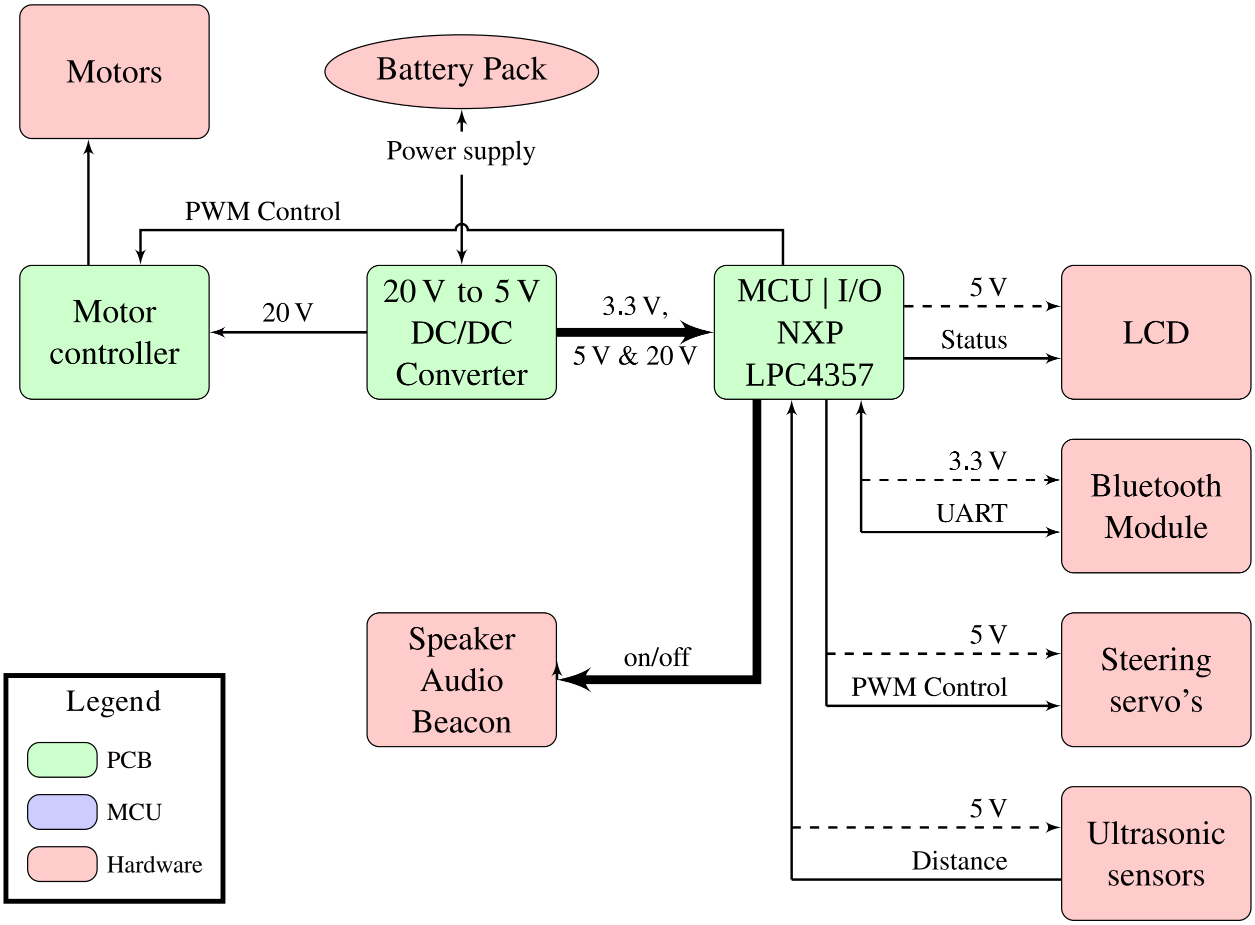


Figure 2: A schematic overview of the entire system in KITT.

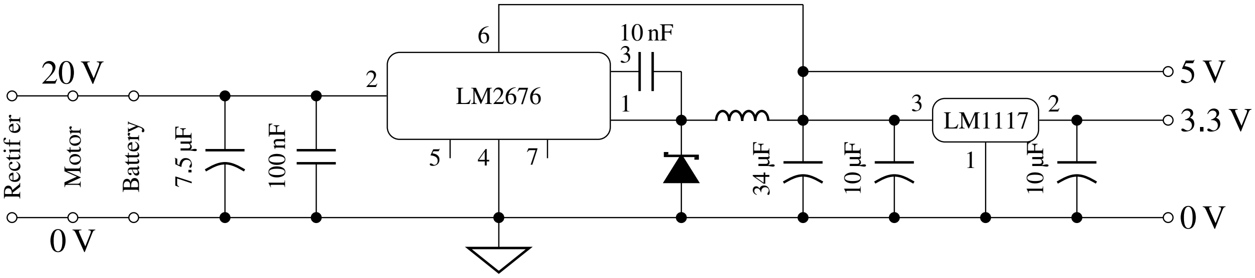


Figure 3: Schematic of the 20V to 5V DC/DC converter

4.1 20 V to 5 V DC-DC converter

This board converts the 20V battery voltage to 5V for the PWM signals to the steering motors and the motor controller, the ultrasonic sensors, the LCD and the microcontroller used for the audio beacon. This is done with a Texas Instruments LM2676 step-down converter as shown in Figure 3. The 5 V is converted to 3.3 V for the master microcontroller and the Bluetooth-module.

The connectors to the DC/DC converter are listed in Table 3.

Table 3: DC/DC Converter I/O

I/O Name	Function
Vin	Input voltage from AC-DC inverter board
Accu	Connected to the battery
Motor	Connected to the drive motor controller
MCU	Outputs supply voltage for the MCU (5 V, GND, 3.3V and 20 V)

4.2 Motor and Motor controller

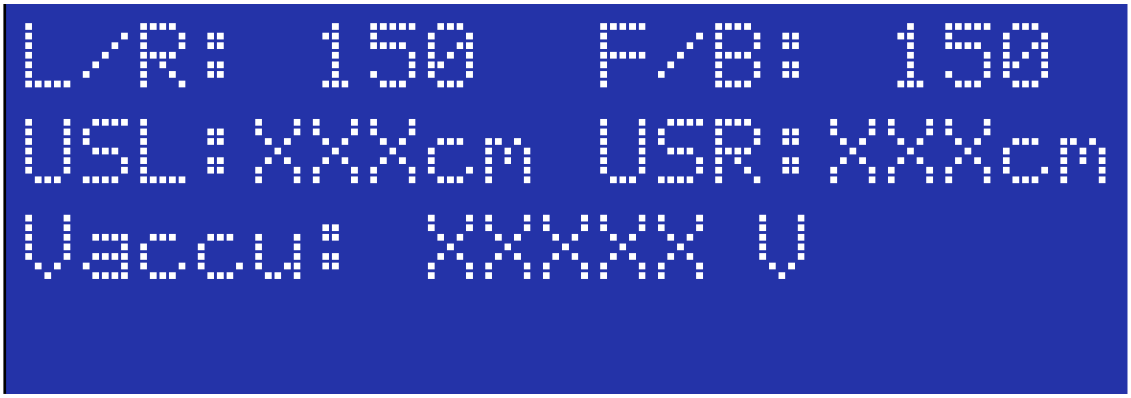
The motor controller amplifies the PWM signal and drives the motors. The motor controller is powerful and therefore must be handled with care. The maximum input voltage is 20 V. The motors are connected parralel and have a maximum continuous voltage of 14 V.

4.3 MCU board

The main controller of KITT is a 32-bit M4 microcontroller, one of the processors in the dual core ARM processor (NXP LPC4357). It is pre-programmed and controls the peripherals on the car.

4.4 Audio Beacon and Speaker

A second M0 microcontroller in the ARM processor is used to control the audio beacon signal. It is preprogrammed, but configurable.



```
L/R: 150  F/B: 150
USL:XXXcm USR:XXXcm
Vaccu: XXXXX V
```

Figure 4: Status of the car on the LCD.

4.5 LCD

The car is equipped with a LCD to monitor the status of the car. In Figure 4) an example of the display in use is shown. The first line contains the drive commands, the second line the ultrasonic sensor data, the third line the battery voltage and the last one the status of the master MCU.

4.6 Bluetooth Module

The car is equipped with a Bluetooth module. It is preconfigured and may not be changed. To stablish a connection, Bluetooth USB modules are available to use with the PC's in the Tellegen Hall.

4.7 Steering-gear

The steering-gear of the car contains two servo motors driven by a PWM signal from the master MCU. The servo's are fed by a 5 V source and controlled with a PWM signal between 10% for complete left and 20% for complete right steering, 15% is straight.

4.8 Ultrasonic sensors

Information about the ultrasonic sensors can be found in it's own datasheet.

5 Bluetooth Communication

The car is equipped with a Bluetooth module as described in Subsection 4.6. The car is controlled only with signals over the Bluetooth channel. There are several functions that can be sent to the car via Bluetooth. Most of them are control commands, some of them are status commands. All commands start with a command type indicator followed by some variables. The types and variables are described in this chapter.

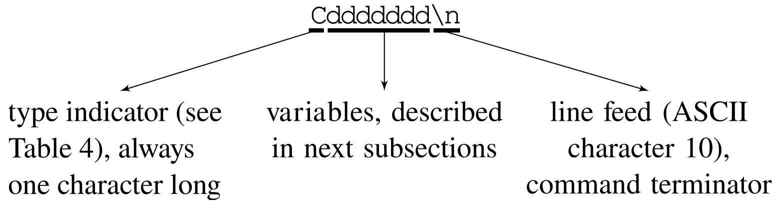

Cddddddd\n
type indicator (see Table 4), always one character long
variables, described in next subsections
line feed (ASCII character 10), command terminator

Figure 5: Command structure.

5.1 Commands

Commands are build-up of nine characters in the form:

In Table 4 the different type indicators are listed. In Table 5 the detailed commands are given.

Table 4: Command Structure

Type indicator	**Type**	**# of arguments**
A	Audio enable	1
B	Set bit frequency	1
C	Set code word	4
D	Set direction PWM	1
F	Set carrier frequency	1
M	Set motor PWM	1
R	Set repetition count	1
S	Get Status	0 or 1
V	Get version info	0

Important note: When the get status command is applied with/without an additional qualifier character, the following status string will be sent to the terminal:

```
Format (x == NULL):
**************************
* Audio Beacon: on
* c: 0x00000000
* f_c: xxxxx
* f_b: xxxxx
* c_r: xxx
**************************
* PWM:
* Dir. xxx
* Mot. xxx
**************************
* Sensors:
* Dist. L xxx R xxx
* V_batt xx.x V
```

```
**************************
Format (x == ’d’):
USL xxx
USR xxx
Format (x == ’v’):
VBATT xx.x V
```

In this status string the PWM promillages, the ultrasonic sensor data, the battery voltage and the audio beacon status will be returned. There are some built in error codes in the returned distance value. When the distance is 999, then it is overloaded.

Table 5: Commands

A:	Audio enable/disable	
“Ax\n”		
x	1:	enable
	else:	disable
Enable/Disable the audio beacon.		
B:	Change bit frequency	
“Bxx\n”		
xx	uint16_t	bit frequency (no characters, but values)
	Default value:	5000 [Hz]
Part of the audio beacon settings:	frequency of bits of audio beacon code.	
C:	Change audio code	
““Cxxxx\n”		
xxxx	uint32_t	32 bits audio code (no characters, but values: 4x 8bit, MSB first)
	Default value:	0x00000000
Part of the audio beacon settings:	frequency of bits of audio beacon code.	
D:	Change direction PWM	
“Dxxx\n”		
xxx	char[3]	PWM value direction (promille), characters
	Min-Max value:	100-200
	Default value:	150
F:	Change carrier frequency	
“Fxx\n”		
xx	uint16_t	carrier frequency (no characters, but values)
	Default value:	15000 [Hz]
Part of the audio beacon settings:	the carrier frequency of the audio beacon.	
M:	Change motor PWM	
“Mxx\n”		
xxx	char[3]	PWM value motor (promille), characters
	Min-Max value:	135-165
	Default value:	150
R:	Change repetition counter	
“Rxx\n”		
xx	uint16_t	repetition counter (no character, but value)
	Min value:	32 (Otherwise a new code is transmitted before the previous one finished)
	Default value:	25000 [Hz]
Part of the audio beacon settings:	the repetition counter.	
The value of this counter is the number of bits the beacon waits to transmit the code again.		
The repetition_frequency = bit_frequency/repetition_counter.		
S:	Get status	
“Sx\n”		
x	char	‘v’: Battery voltage ‘d’: Distance ‘NULL’: all
	Return:	Status of Kitt